The Slack World

Issue #1, April 2005

SlackTips

Author: Mikhail Zotov

The section is aimed at new Slackware Linux users and intends to help them make their life in Linux more effective. In this issue, we'll take a look how aliases and functions can speed up some everyday operations in bash.

First, a quote from **man bash**:

> When **bash** is invoked as an interactive login shell, or as a non-interactive shell with the **--login** option, it first reads and executes commands from the `/etc/profile`, if that file exists. After reading that file, it looks for `~/.bash_profile`, `~/.bash_login`, and `~/.profile`, in that order, and reads and executes commands from the first one that exists and is readable ... When an interactive shell that is not a login shell is started, **bash** reads and executes commands from `~/.bashrc`, if that file exists.

In a freshly installed Slackware system, users have none of these files in their home directories. Thus we first need to create them:

```
$ touch ~/.bash_profile ~/.bashrc
```

We want our aliases be read every time an interective shell is started (e.g., when we start **xterm**) and will thus use `~/.bashrc` for our purpose. To ensure that it is read every time we start an intercative shell, let's put the following lines in `~/.bash_profile`:

```
# .bash_profile
if [ -f ~/.bashrc ]; then
  . ~/.bashrc
fi
```

From now on, I assume that all aliases and functions are put in `~/.bashrc`.

The first thing to keep in mind is that every time we add something to `~/.bashrc` we should make the shell know about the changes. Thus we must **source** the file (see **man bash** or **help source**). To do this, one may run either

```
$ source ~/.bashrc
```

or

```
$ . ~/.bashrc
```

Let's avoid doing this manually.

Suppose your favourite editor is **mcedit**. Let's open `~/.bashrc` with it and put the following **alias** [1]:

```
alias edb='mcedit ~/.bashrc && . ~/.bashrc'
```

Now, exit **mcedit** and **source** `~/.bashrc`. From now on, the only thing we need to do when we decide to add a new alias to `~/.bashrc` and make the shell know about it, is to run

```
$ edb
```

Here I use **edb** as an abbreviation for EDit `~/.Bashrc` [2].

Now we are ready to do something useful. Let's begin with a trivial task, namely, let us mount and unmount a floppy. As is well known, this can be done as follows:

```
$ mount /mnt/floppy
$ umount /mnt/floppy
```

These commands look too long for an everyday usage. Yes, one can surely use <Ctrl>+<r> or arrow keys or **history** to invoke commands that has already been run but I believe it is more convenient to use short aliases in this case. Let us run **edb** once again and add the following lines to ~/.bashrc:

```
# Mount Floppy
alias mf='mount /mnt/floppy && cd /mnt/floppy && ls'
# Unmount Floppy
alias uf='cd && umount /mnt/floppy'
```

Exit from the editor, insert a floppy in the drive, and try our brand new aliases. Notice that we not only mount the floppy but also **cd** to it and list its contents. This can be handy.

Using CDs calls us to invent more aliases since we can open and close trays from the command line. Let's put the following aliases in ~/.bashrc (once again, **$ edb**):

```
CDROM="/mnt/cdrom"
alias mcd='mount $CDROM && cd $CDROM && ls'
alias ucd='cd && umount $CDROM && eject && sleep 10 && eject -t'
# Eject CD:
alias ecd='eject'
# Close CD tray:
alias ccd='eject -t'
```

Notice that now we not just **umount** a CD but also eject it, give ourselves 10 seconds to take it, and then close the tray automatically. Besides this, we define a variable, CDROM. It can be useful if one day we decide to use another mount point.

Here I assume that we only have one CD drive. During installation, Slackware creates the corresponding link in the /dev directory. You will have to adopt the above aliases in case there are two CD drives attached to your machine. In particular, you will have to indicate explicitly the second device in **eject**.

Now, assuming that we have a CD-RW drive, let's do something interesting. Yes, let's burn a disk. Everybody surely knows how to burn a CD 'at once' from an image downloaded from a Slackware mirror. This doesn't happen too often thus one can forget how this is done. Let's define an alias for this operation, e.g., this way:

```
DEV="dev=0,0,0"
alias burn='ccd && cdrecord -eject $DEV -dao'
```

Here, "0,0,0" is taken from the output of **cdrecord -scanbus**. (One may want to add, say **speed=16** or whatever to be sure that the drive will burn CDs at the desired speed.) We also define another variable, DEV, which will be used below.

Now, to burn a Slackware CD, one only has to eject the CD tray (**ecd**), put a blank CD, and execute the following command:

```
$ burn /path/to/the/image/slackware-10.1-install-d1.iso
```

Voila! Notice that we don't even have to close the CD tray.

Let's see now how we can easily burn multi-session CDs from the CLI. We shall use bash functions to accomplish the feat.

First, let us begin a new CD. I assume that we are using a CD-RW thus we'll blank it first to be sure it is clean. Next, we shall make an iso image from files prepared in a directory the name of which will be passed as a parameter. The image will be saved in the home directory. Finally, we shall burn the CD, eject it, and delete the iso image.

```
# iso image:
```

```
ISO="/tmp/a.iso"

# Make the iso image:
alias mkiso='mkisofs -R -J -v -hide-rr-moved -o $ISO'

# Time cdrecord waits before burning a CD
WAIT="gracetime=5"

# Another handy alias:
alias BURN="cdrecord -v -eject $DEV $WAIT -tao -multi $ISO && \
  rm -f $ISO && sleep 10 ; ccd"

begincd() {
  ccd && \
  cdrecord -v blank=fast $DEV $WAIT && \
  mkiso $1 && BURN
}

addtocd() {
  ccd && \
  mkiso -C `cdrecord -msinfo $DEV` -M /dev/cdrom $1 && \
  BURN
}
```

A few comments are in order. First, we define a file to be used as an iso image (`ISO`). Next, we define an alias that will not only save us some keystrokes in the next two functions but can also be used when we just need to create an iso image, e.g., in case we want to burn a complete CD of our own. Besides this, we introduce a variable `WAIT`, which will save us another couple of seconds. (We are in a hurry, aren't we?) Finally, we define another alias, `BURN`. It will be only used in our two functions thus I choose to put its name in capitals.

Thus, to begin a CD with files from the `dir1` directory, one now only has to eject the CD tray (**ecd**), put a CD-RW, and execute the following command:

```
$ begincd dir1
```

Similarly, to add files from `dir2` to the CD, one ejects the CD tray, puts the CD, and executes the command

```
$ addtocd dir2
```

Quick and easy, isn't it? Actually, we can even put **ecd** at the beginning of our definitions and then **sleep** for some time. `:-)`

As we have seen, aliases and functions are powerful tools. They can be employed to do numerous different things:

- to navigate directories:

  ```
  alias cda='cd ~/some/directory/where/I/am/writing/an/Article'
  ```

- to pack and unpack tar balls:

  ```
  alias tgz='tar czpvf'
  alias utgz='tar xzvf'
  ```

  as easy as:

  ```
  $ tgz arxiv.tgz slaxercises/
  $ utgz arxiv.tgz
  ```

- to backup files, say, configuration files:

```
BACKUP_DIR="/path/to/the/backup/directory"
alias back="cd $BACKUP_DIR && tgz conf-‘date +%F‘.tgz ~/.??* && \
  ls && cd"
```

- to keep a record of upgraded and removed packages (surely, these are for root's `.bashrc`) [3]:

```
UPLOG="~/upgradepkg.log"
uplog() {
  date >> $UPLOG
  upgradepkg $@ | tee -a $UPLOG
}

REMLOG="~/removepkg.log"
remlog() {
  date >> $REMLOG
  removepkg $@ | tee -a $REMLOG
}
```

- to verify PGP signatures of freshly downloaded Slackware packages:

```
alias gpgv="for i in ./*.tgz ; do echo $i ; \
  gpg --no-secmem-warning --verify $i.asc $i ; echo ; done"
```

- to listen music, e.g., to play all mp3 or wav files in the current directory:

```
alias mp3='madplay -v --display-time=current ./*.mp3'
alias wav='for i in ./*.wav ; do play $i ; done'
```

- watch a collection of, say, jpg pics:

```
alias slide='qiv -f -s --delay=5 -i ./*.jpg'
```

and do dozens of other things that make using Slackware even more fun than it is. Use your imagination!

Remarks

[1] A more generic way is the following:

```
alias edb='$VISUAL ~/.bashrc && . ~/.bashrc'
```

I suggest that you check first whether `$VISUAL` really points to the desired editor ($ **declare** | **grep** `VISUAL`) in order not to find yourself playing with **elvis** unintentionally. ;-)

[2] Feel free to choose another one. The only thing one should check before inventing a new name is to check that it is not already occupied by a shell built-in or a program in your PATH. To do this, one can run

```
$ help new_name
$ which new_name
```

The **new_name** is free if you get no help and nothing is found.

[3] These records can, in particular, be used to check whether this or that package left any undeleted files after being removed or upgraded.

The Slack World <http://www.slackworld.net/> invites you to send us your tips for the benefit of the Slackware community.